# MAC0325/5781 COMBINATORIAL OPTIMIZATION: ASSIGNMENT 3

**Submit:** Exercises 1, 3, 14, and 18
**Deadline:** 14/Dec/2020
**Instructions:** Follow **CAREFULLY** the submission instructions listed in Moodle.

## 1. Flows, Transshipments, and Circulations

**Exercise 1** (Reduction from Shortest Walk to Min-Cost Flow)**.** Let $D = (V, A, \varphi)$ be a digraph, and let $c\colon A \to \mathbb{R}$. Let $r, s \in V$. Suppose that $r \rightsquigarrow v \rightsquigarrow s$ for each $v \in V$. Construct an instance $(D, c, u, r, s, \beta)$ of the Min-Cost Flow Problem (*not* the integral version!) such that the Shortest Walk Problem (SWP) on $(D, c, r, s)$ is (homomorphically) equivalent to (18.16) from the lecture notes on $(D, c, u, r, s, \beta)$.

*Hint.* In order to send a feasible solution $f$ for (18.16) (that is not necessarily integral) to a feasible solution for (SWP) with at least as good objective value, rely on Exercise 18.10 from the lecture notes. If one has $f = \sum_{P \in \mathcal{P}} x_P \mathbb{1}_P$, where $\mathcal{P}$ is a set of $rs$-paths in $D$, and $x \in \mathbb{R}_+^{\mathcal{P}}$ is a vector such that $\mathbb{1}^\mathsf{T} x = 1$, can one have $c^\mathsf{T} f < c(P)$ for each $P \in \mathcal{P}$?

**Exercise 2** (Reduction from Bipartite Min-Cost Perfect Matching to Min-Cost Flow)**.** Let $G = (V, E, \varphi)$ be a $(U, W)$-bipartite graph such that $|U| = |W|$, and let $c\colon E \to \mathbb{R}$. Construct an instance $(D, c, u, r, s, \beta)$ of the Min-Cost Integral Flow Problem such that the Minimum-Cost Perfect Matching Problem (19.1) (from the lecture notes) on $(G, U, W, c)$ is isomorphic to (18.17) from the lecture notes on $(D, c, u, r, s, \beta)$. Can you also prove (homomorphic) equivalence to the non-integral version, i.e., to (18.16)?

**Exercise 3** (Reduction from Min-Cost Flow to Min-Cost Circulation)**.** Let $D = (V, A, \varphi)$ be a digraph. Let $c\colon A \to \mathbb{R}$. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let $r, s \in V$ be distinct. Let $\beta \in \mathbb{R}_+$. Construct an instance $(D', c', \ell', u')$ of the Min-Cost Circulation Problem such that the Min-Cost Flow Problem (18.16) on $(D, c, u, r, s, \beta)$ is isomorphic to $(\mathrm{MCC}(D', c', \ell', u))$. The (appropriate restriction of the) isomorphism must also be an isomorphism between the integral versions of the problems, when the parameters $u$ and $\beta$ are integral.

**Exercise 4** (Reduction from Min-Cost Transshipment to Min-Cost Flow)**.** Let $D = (V, A, \varphi)$ be a digraph. Let $c\colon A \to \mathbb{R}$. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let $b \in \mathbb{R}^V$ such that $\mathbb{1}^\mathsf{T} b = 0$. Construct an instance $(D', c', u', r', s', \beta')$ of the Min-Cost Flow Problem (18.16) (of the lecture notes) such that the Min-Cost Transshipment Problem (18.18) on $(D, c, u, b)$ is isomorphic to (18.16) on $(D', c', u', r', s', \beta')$. The (appropriate restriction of the) isomorphism must also be an isomorphism between the integral versions of the problems, when the parameters $u$ and $b$ are integral.

**Exercise 5** (Reduction from Min-Cost Circulation to Min-Cost Transshipment)**.** Let $D = (V, A, \varphi)$ be a digraph. Let $c\colon A \to \mathbb{R}$. Let $\ell\colon A \to \mathbb{R}_+$ be a vector. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Construct an instance $(D', c', u', b')$ of the Min-Cost Transshipment Problem (18.18) (of the lecture notes) and a linear-time algorithm that, given any optimal solution for (18.18) applied to $(D', c', u', b')$, returns an optimal solution for $(\mathrm{MCC}(D, c, \ell, u))$. The same construction (and algorithm) must also work for the integral versions of the problems, when the parameters $\ell$ and $u$ are integral.

---

*Date*: November 13, 2020 at 01:31pm -0300.

**Exercise 6** (Reduction of Maximum Flow to Min-Cost Flow)**.** Consider Algorithm 1 for an instance $(D, u, r, s)$ of the Maximum Flow problem. Design *linear-time algorithms* PREPROCESS and POSTPROCESS with the following properties. PREPROCESS takes as input an instance $(D, u, r, s)$ of the Maximum Flow problem and returns either the outcome "unbounded" or an instance $(D', c', u', r', s', \beta')$ of the Min-Cost Flow Problem (18.16) (from the lecture notes) that has an optimal solution. The (black-box) function MINCOSTFLOW takes as input an instance of the Min-Cost Flow Problem that has an optimal solution and returns an optimal solution. POSTPROCESS takes as input an optimal solution for Min-Cost Flow Problem and returns an optimal solution for Maximum Flow on input $(D, u, r, s)$.

**Algorithm 1** Maximum Flow via Single Min-Cost Flow

```
1: function MAX-FLOW(D, u, r, s)
2:     Ret ← PREPROCESS(D, u, r, s)
3:     if Ret is unbounded then
4:         return unbounded
5:     (D', c', u', r', s', β') ← Ret
6:     f' ← MINCOSTFLOW(D', c', u', r', s', β')
7:     return POSTPROCESS(f')
```

**Exercise 7** (Bootstrapped/2-Phase Min-Cost Circulation Algorithm)**.** Write two algorithms, MINCOSTCIRCULATIONFROMFEASIBLE and MINCOSTCIRCULATION with the following properties. Algorithm MINCOSTCIRCULATIONFROMFEASIBLE takes as input an instance $(D, c, \ell, u)$ of the Min-Cost Circulation Problem (MCC$(D, c, \ell, u)$) *and* a feasible solution for the optimization problem, and it returns either "unbounded" and a certificate of unboundedness, as in Algorithm 21.1 from the lecture notes, or an optimal solution for the optimization problem. Algorithm MINCOSTCIRCULATION takes as input an instance $(D, c, \ell, u)$ of the Min-Cost Circulation Problem (MCC$(D, c, \ell, u)$) and makes *exactly two calls* to Algorithm MINCOSTCIRCULATIONFROMFEASIBLE, and solves the given instance of the Min-Cost Circulation Problem. No calls to other algorithms are allowed.

*Hint.* The first call should use an instance of the Min-Cost Circulation that is has an "obvious" feasible solution.

**Exercise 8.** Let $D \coloneqq (V, A, \varphi)$ be a digraph. A trail $W$ in $D$ is *Eulerian* if $A(W) = A$. It is not hard to prove that, if the underlying digraph of $D$ is connected, then $D$ has a closed Eulerian trail if and only if $|\delta^{\text{in}}(v)| = |\delta^{\text{out}}(v)|$ for each $v \in V$. Let $k\colon A \to \mathbb{R}_+$ be a cost vector. Recall the notation $\widehat{D} = (V, \widehat{A}, \widehat{\varphi})$ and from §20.7 from the lecture notes. For each $B \subseteq A$, denote the digraph $D_B \coloneqq (V, B', \widehat{\varphi}\restriction_{B'})$, where $\overline{B} \coloneqq A \setminus B$ and

$$B' \coloneqq (\overline{B} \times \{+1\}) \cup (B \times \{-1\}).$$

Design an efficient algorithm for the following optimization problem

$$\begin{aligned} &\text{Minimize} && k(B) \\ &\text{subject to} && B \subseteq A \\ &&& D_B \text{ has a closed Eulerian trail.} \end{aligned} \tag{8.1}$$

**Exercise 9.** Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $k\colon A \to \mathbb{R}_+$ be a cost vector. Define the digraph $D_{\mathbb{N}} \coloneqq (V, A_{\mathbb{N}}, \varphi_{\mathbb{N}})$ by setting $A_{\mathbb{N}} \coloneqq A \times \mathbb{N}$ and $\varphi_{\mathbb{N}}\big((a, i)\big) \coloneqq \varphi(a)$ for each $(a, i) \in A_{\mathbb{N}}$. For each $m \in \mathbb{N}^A$, denote the digraph $D_m \coloneqq (V, A_m, \varphi_{\mathbb{N}}\restriction_{A_m})$, where

$$A_m \coloneqq \{\, (a, i) : a \in A,\, i \in [m_a] \}.$$

Design an efficient algorithm for the following optimization problem

$$\begin{aligned} &\text{Minimize} && k^{\mathsf{T}} m \\ &\text{subject to} && m \in \mathbb{N}^A, \\ &&& D_m \text{ has a closed Eulerian trail.} \end{aligned} \tag{9.1}$$

**Exercise 10.** Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $f \in \mathbb{R}_+^A$. Formulate the following optimization problem as a Minimum Cost Integral Flow Problem:

$$\begin{aligned} &\text{Minimize} && \|f - g\|_1 \\ &\text{subject to} && g \in \mathbb{Z}_+^A, \\ &&& \text{excess}_f = \text{excess}_g, \\ &&& \lfloor f \rfloor \leq g \leq \lceil f \rceil. \end{aligned} \tag{10.1}$$

Here, the 1-norm of a vector $x \in \mathbb{R}^S$ is $\|x\|_1 \coloneqq \sum_{s \in S} |x_s|$, and the functions $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ are applied to vectors componentwise.

## 2. Matchings

**Exercise 11** (Reduction from Max-Weight Matching to Min-Cost Perfect Matching)**.** The **maximum-weight matching problem** is, **given**

(i) a graph $G = (V, E, \varphi)$,
(ii) a weight function $w \colon E \to \mathbb{R}$,

solve the optimization problem

$$\begin{aligned} &\text{Maximize} && w(M) \\ &\text{subject to} && M \text{ is a matching in } G. \end{aligned} \tag{11.1}$$

Let $(G, w)$ be an instance of the maximum-weight matching problem. Build an instance $(G', c)$ of the minimum-cost perfect matching as follows. Set $G' \coloneqq (V', E', \varphi')$, where $V' \coloneqq V \sqcup V$, $E' \coloneqq E \sqcup E \sqcup V$, and

$$\varphi'((i,x)) \coloneqq \begin{cases} \{(u,i),(v,i)\} & \text{if } i \in [2] \text{ (whence } x \in E\text{), where } \{u,v\} \coloneqq \varphi(x), \\ \{(v,1),(v,2)\} & \text{if } i = 3 \text{ (whence } x \in V\text{), where } v \coloneqq x, \end{cases}$$

for each $(i,x) \in E'$. Define $c$ and a linear-time algorithm $\mathcal{A}$ such that, if $M'$ is an optimal solution for the Min-Cost Perfect Matching Problem (19.1) (from the lecture notes) on $(G', c)$, then $\mathcal{A}(M')$ (the return value of a call to $\mathcal{A}$ with parameter $M'$) is an optimal solution for (11.1) on $(G, w)$. When $G$ is bipartite, is $G'$ bipartite as well? Prove this or provide a counterexample.

**Exercise 12** (Reduction from Min-Cost Perfect Matching to Max-Weight Matching)**.** Let $(G, c)$ be an instance of the minimum-cost perfect matching problem. Define an instance $(G, w)$ of the maximum-weight matching problem (11.1) such that, if $M$ is an optimal solution for (11.1) on $(G, w)$, then $M$ is an optimal solution for the Min-Cost Perfect Matching Problem (19.1) (from the lecture notes) on $(G, c)$.

*Attribution.* This reduction is taken from [1, Prop. 11.1].

**Exercise 13.** Run Pap's Algorithm to find a maximum matching and a certificate of optimality (as in the Tutte-Berge Formula) for the graph in Figure 1.

**Exercise 14.** Run the Hungarian Method to find a minimum-cost perfect matching and a certificate of optimality (an optimal solution for the dual LP (19.5) from the lecture notes) for the following input. Set $V \coloneqq [10]$, $U \coloneqq [5]$, and $W \coloneqq [10] \setminus U$. Set $E \coloneqq \{\, uw : u \in U,\, w \in W \}$ and $G \coloneqq (V, E)$. The cost of each edge $uw \in E$ is provided by Table 1. You must follow **precisely** the instructions below.

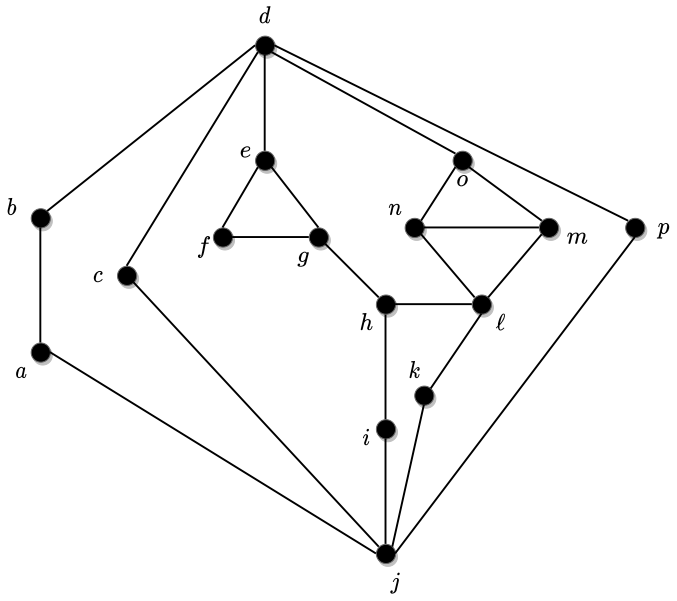


FIGURE 1. The input instance for Exercise 13.

| | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|
| 1 | 6 | 17 | 10 | 1 | 3 |
| 2 | 9 | 23 | 21 | 4 | 5 |
| 3 | 2 | 8 | 5 | 0 | 1 |
| 4 | 19 | 31 | 19 | 20 | 9 |
| 5 | 21 | 25 | 22 | 3 | 9 |

TABLE 1. Cost of edges of $G$.

*Instructions.* For each iteration of the **for** loop in the Hungarian Method, you must write which one of current matching and dual solution actually changes.

In the case of a matching/primal update, you must always find a **minimum-length** $M_t$-augmenting path from a vertex in $U \setminus V_{M_t}$ to a vertex in $W \setminus V_{M_t}$ and, among all such minimum-length augmenting paths, you must choose **the lexicographically smallest** one. You should write the augmenting path (skipping edges is fine) used and the updated matching $M_{t+1}$.

In the case of a dual update, specify the vertex cover $K_t$, the direction vector $d_t$, the step size $\lambda_t$, and the new dual solution $y_{t+1}$.

When writing vectors indexed by the vertex set [10], it is fine to write the vector as a 10-tuple.

**Exercise 15.** Derive Kőnig's Matching Theorem (Corollary 8.8) from the Tutte-Berge Formula.

**Exercise 16** (CCPS Ex. 5.7)**.** Let $G \coloneqq (V, E)$ be a 3-regular graph. Suppose that for each edge $e \in E$, the graph $(V, E \setminus \{e\})$ is connected. Prove that $G$ has a perfect matching.

**Exercise 17** (CCPS Ex. 5.18, attributed to W. Anderson)**.** *Slither* is a two-person game played on a graph $G \coloneqq (V, E)$. The players, called *First* and *Second*, play alternately, with First playing first. At each step, the player whose turn it is chooses a previously unchosen edge. The only rule is that at every step the set of chosen edges is the set of edges of a path. The loser is the first player unable to make a legal play at his or her turn. Prove that, if $G$ has a perfect matching, then First can force a win.

**Exercise 18.** Let $G \coloneqq (V, E, \varphi)$ be a graph. Call a vertex $v \in V$ *essential* (in $G$) if, for each maximum matching $M$ of $G$, one has $v \in V_M$; equivalently, if $\nu(G - v) < \nu(G)$. Suppose that $G$

has no essential vertices. Define the binary relation $\sim$ on $V$ by setting $u \sim v$ if $u = v$ or ($u \neq v$ and there is no maximum matching $M$ of $G$ such that $u$ and $v$ are $M$-exposed). Note that the condition in parentheses can be rewritten as: for every maximum matching $M$ of $G$, we have $\{u, v\} \cap V_M \neq \varnothing$. Prove that $\sim$ is an equivalence relation.